

**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук
Образовательная программа «Программная инженерия»

СОГЛАСОВАНО

УТВЕРЖДАЮ

Технический руководитель
внештатный преподаватель департамента
программной инженерии

Академический руководитель образовательной
программы «Программная инженерия»
профессор департамента программной
инженерии, канд. техн. наук



Н. И. Веселко
«10» апреля 2023 г.

В. В. Шилов
«10» апреля 2023 г.

**МОДУЛЬ УТИЛИТАРНЫХ АБСТРАКЦИЙ МОДЕЛЕЙ ДЛЯ ФРЕЙМВОРКА
DJANGO**

Руководство программиста

ЛИСТ УТВЕРЖДЕНИЯ

RU.17701729.04.03-01 33 01-1-ЛУ

Подп. и дата	
Инв. № дубл.	
Взам. инв. №	
Подп. и дата	
Инв. № подл	

Исполнитель
студент группы БПИ203



Н. С. Черемин
«10» апреля 2023 г.

Москва 2023

УТВЕРЖДЕН
RU.17701729.04.03-01 33 01-1-ЛУ

**МОДУЛЬ УТИЛИТАРНЫХ АБСТРАКЦИЙ МОДЕЛЕЙ ДЛЯ ФРЕЙМВОРКА
DJANGO**

Руководство программиста

RU.17701729.04.03-01 33 01-1

Листов 20

Подп. и дата	
Инв. № дубл.	
Взам. инв. №	
Подп. и дата	
Инв. № подл	

Москва 2023

СОДЕРЖАНИЕ

ГЛОССАРИЙ.....	4
АННОТАЦИЯ.....	5
1. НАЗНАЧЕНИЕ И УСЛОВИЯ ПРИМЕНЕНИЯ ПРОГРАММ.....	6
1.1. Назначение программы.....	6
1.2. Функции, выполняемые программой.....	6
1.2.1. BaseModel.....	6
1.2.2. TimedModel.....	6
1.2.3. DeletableModel.....	7
1.2.4. AutoDeletableModel.....	7
1.2.5. LogModel.....	8
1.2.6. ContinuousLogModel.....	8
1.2.7. UpdateLogModel.....	8
1.2.8. UpdatableModel.....	8
1.2.9. UpdatableLoggableModel.....	10
1.2.10. HistoryModel.....	12
1.3. Условия, необходимые для выполнения программы.....	13
2. ХАРАКТЕРИСТИКА ПРОГРАММЫ.....	14
3. ОБРАЩЕНИЕ К ПРОГРАММЕ.....	15
4. ВХОДНЫЕ И ВЫХОДНЫЕ ДАННЫЕ.....	16
5. СООБЩЕНИЯ.....	17
СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ.....	18
ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ.....	20

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

ГЛОССАРИЙ

- ПО — программное обеспечение [17].
- БД — база данных [19].
- ООП — объектно-ориентированное программирование [20].
- SDK (software development kit) — набор (пакет) средств (инструментов) разработки ПО [15].
- ORM (object-relational mapping) — технология программирования, которая связывает базы данных с концепциями объектно-ориентированных языков программирования, создавая «виртуальную объектную базу данных» [18].
- Модель (данных) — абстрактное, логическое определение объектов, операторов и прочих элементов, в совокупности составляющих абстрактную машину доступа к данным, с которой взаимодействует пользователь [16]. Здесь: описание набора полей, из которых состоит таблица данных, и их свойств, а также свойств самой таблицы, используемое Django ORM [14].
- Кверисет (здесь) — класс, использующийся для получения записей моделей и наследующий `django.db.models.QuerySet`.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

АННОТАЦИЯ

Данный документ предназначен для описания способов использования программы программистами, работающими с фреймворком Django [14].

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

1. НАЗНАЧЕНИЕ И УСЛОВИЯ ПРИМЕНЕНИЯ ПРОГРАММ

1.1. Назначение программы

Пакет содержит абстрактные модели — Python [21] классы, унаследованные (напрямую или через других родителей) от `django.db.models.Model`. Абстрактные модели предоставляют базовый функционал (переопределяемый наследниками) для моделей логов, моделей логов обновления, обновляемых логируемых моделей, а также исторических моделей.

Пакет предназначен для наследования пользовательскими моделями предоставляемых пакетом абстрактных моделей для упрощения написания кода и единого базового поведения всех наследуемых моделей.

1.2. Функции, выполняемые программой

1.2.1. BaseModel

Модель предоставляет публичную функцию `values(self, *fields, **expressions)`, которая возвращает словарь пар ключ-значение, где ключ — название поля, а значение — фактическое значение этого поля в записи `self`. Данная функция копирует поведение одноименной функции стандартного кверисета доступного через статическое поле `objects` в наследующих `django.models.db.Model` моделях (далее: стандартная модель и стандартный кверисет), только не для списка записей, а для одной записи.

1.2.2. TimedModel

Модель наследует `BaseModel`.

Дополнительные поля модели: `time_created` — время создания записи (значение по умолчанию — текущее время).

Дополнительные публичные атрибуты модели: `ordering` — список названий полей для сортировки записей.

Модель предоставляет публичную функцию `get_ordering(cls, *args, **kwargs)`, которая возвращает атрибут `ordering` (функция добавлена для переопределения при необходимости).

Модель предоставляет публичную функцию `get_last_created_object(cls, *args, **kwargs)` для получения последней по времени создания записи.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

1.2.3. DeletableModel

Модель наследует TimedModel (использует ее функционал для работы со старыми записями).

Дополнительные публичные атрибуты модели: `max_objects_count` — максимально допустимое количество записей.

Модель предоставляет публичную функцию `get_max_objects_count(cls, *args, **kwargs)`, которая возвращает атрибут `max_objects_count` (функция добавлена для переопределения при необходимости).

Модель предоставляет публичную функцию `get_objects_to_delete(cls, *args, **kwargs)`, которая возвращает записи, которые должны быть удалены. Количество таких записей равно разности между максимальным (`max_objects_count`) и текущим количеством записей. Если разность положительна (то есть хранится больше записей, чем допустимо), функция выбирает `x` наиболее старых (по полю `time_created`) записей, где `x` — описанная выше разность.

Модель предоставляет публичную функцию `try_delete_objects(cls, *args, **kwargs)`, которая удаляет записи, полученные из функции `get_objects_to_delete`, при помощи функции `delete` стандартного кверисета и возвращает возвращаемое этой функцией значение.

1.2.4. AutoDeletableModel

Модель наследует DeletableModel (использует ее функционал для автоматической очистки старых записей).

Дополнительные публичные атрибуты модели: `objects` — для переопределения стандартного кверисета специальным (`AutoDeleteQuerySet`).

Кверисет `AutoDeleteQuerySet` переопределяет функцию `bulk_create` стандартного кверисета для вызова исключения при попытке одновременного создания записей, после создания которых количество записей модели будет превышать максимально допустимое (`max_objects_count`).

Модель переопределяет функцию `save` стандартной модели, которая вызывает родительскую функцию `save`, а затем функцию `try_delete_objects`. Переопределение данной функции позволяет поддерживать автоматическое удаление старых записей при каждом

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

создании новой записи, если установлено максимально количество записей (max_objects_count).

1.2.5. LogModel

Модель наследует AutoDeletableModel (использует ее функционал очистки старых записей при соответствующей настройке).

Дополнительные поля модели: messages — поле для хранения в логе сообщений.

1.2.6. ContinuousLogModel

Модель наследует LogModel (расширяет ее для работы с логами продолжительных событий, то есть имеющих не только время создания, но и время окончания, при этом время создания интерпретируется как время начала).

Дополнительные поля модели: time_finished — время окончания события.

1.2.7. UpdateLogModel

Модель наследует ContinuousLogModel (расширяет ее для работы с логами обновлений).

Дополнительные поля модели: was_canceled — флаг для отметки отмены обновления, was_terminated — флаг для отметки прерывания обновления, has_failed — флаг для отметки возникновения ошибки в ходе обновления (все 3 со значениями по умолчанию — ложь)

1.2.8. UpdatableModel

Модель наследует BaseModel.

Дополнительные публичные атрибуты модели: time_limit — экземпляр datetime.timedelta для настройки предельного времени обновления (при его превышении процесс обновления может быть прерван).

Дополнительные защищенные атрибуты модели: _update_lock — экземпляр threading.Lock (для того, чтобы контролировать, что в любой момент времени может протекать только один процесс обновления), _check_lock — также экземпляр threading.Lock (для контроля того, чтобы контролировать, что в любой момент времени может осуществляться только одна попытка старта нового обновления, _time_started — время начала текущего обновления, _must_terminate — флаг для уведомления о необходимости прерывания текущего обновления.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

Модель предоставляет публичную функцию `get_time_limit(cls, *args, **kwargs)`, которая возвращает атрибут `time_limit` (функция добавлена для переопределения при необходимости).

Модель предоставляет публичную функцию `is_updating(cls, *args, **kwargs)`, которая возвращает истину, если `_update_lock` заблокирован (то есть если имеется текущее обновление).

Модель предоставляет публичную функцию `is_can_start(cls, *args, **kwargs)`, которая возвращает истину, если в данный момент нет текущего обновления (функция `is_updating`) или если время текущего обновления (вычисляется с помощью `_time_started`) превышает предельное (`time_limit`). Иначе — возвращает ложь. Таким образом, функция определяет, можно ли начать новое обновления или нет.

Модель предоставляет защищенную функцию `_handle_cannot_start(cls, *args, **kwargs)`, которая в данной абстракции ничего не делает (функция добавлена для переопределения при необходимости). Предполагается, что функция будет содержать код реагирования на невозможность начать новое обновление.

Модель предоставляет защищенную функцию `_pre_update(cls, *args, **kwargs)`, которая в данной абстракции ничего не делает (функция добавлена для переопределения при необходимости). Предполагается, что функция будет содержать код подготовки обновления

Модель предоставляет защищенную функцию `_update(cls, *args, **kwargs)`, которая в данной абстракции ничего не делает (функция добавлена для переопределения при необходимости). Предполагается, что функция будет содержать код самого обновления.

Модель предоставляет защищенную функцию `_post_update(cls, *args, **kwargs)`, которая в данной абстракции ничего не делает (функция добавлена для переопределения при необходимости). Предполагается, что функция будет содержать код действий сразу после обновления.

Модель предоставляет защищенную функцию `_handle_exception(cls, exception, *args, **kwargs)`, которая в данной абстракции ничего не делает (функция добавлена для переопределения при необходимости). Предполагается, что функция будет содержать код реагирования на возникновение ошибки в ходе обновления.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

Модель предоставляет публичную функцию `update(cls, *args, **kwargs)`, которая осуществляет обновление. Функция сразу блокирует `_check_lock` для последующей попытки старта обновления (это нужно для того, чтобы не получилось так, что 2 и более параллельных вызова функции не получили разрешение на обновление одновременно). Далее функция определяет, может ли быть начато новое обновление (вызовом функции `is_can_start`). Если не может, вызывается функция `_handle_cannot_start` и `_check_lock` разблокируется, а сама функция завершается. Иначе, устанавливается значения истина в атрибут `_must_terminate` (флаг, который должен отслеживаться в реализациях функции `_update`), после чего функция ожидает разблокировки `_update_lock`. Как только это происходит, функция устанавливается значение ложь в атрибут `_must_terminate`, сохраняет время начала обновления (`_time_started`) и блокирует `_update_lock` (это нужно, чтобы даже если другой вызов получит разрешение на новое обновление, оно не началось, пока данное выполнение не завершится, при этом ответственность за принудительное завершение лежит на реализации функции `_update`). Далее в блоке `try-except` (для предотвращения выбрасывания любых исключений) вызываются переопределяемые в реализациях функции `_pre_update`, `update` и `_post_update`. При возникновении исключения, вызывается также в блоке `try-except` (тоже для предотвращения выбрасывания любых исключений) вызывает переопределяемая в реализациях функция `_handle_exception`. При возникновении исключения в этом блоке, ничего не происходит (предотвращение выбрасывания). Идея в том, чтобы ни одно обновление никогда не выкидывало ошибку и не было необходимости в дополнительных мерах безопасности. Функция возвращает 3 значения: первое: истину, если было успешно начато и завершено без ошибок, иначе — ложь, второе: ошибку, если таковая возникла в первом блоке `try-except` (то есть в ходе обновления), третье: ошибку, если таковая возникла во втором блоке `try-except` (то есть в ходе попытки отреагировать на ошибку обновления).

1.2.9. UpdatableLoggableModel

Модель наследует `UpdatableModel` (для добавления к ее функционалу логирования обновлений).

Дополнительные публичные атрибуты модели: `log_model` — модель лога (предполагается, что наследник `UpdateLogModel`), `use_average_time` — флаг для расчета среднего времени обновления при проверке возможности прерывания текущего обновления

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

(при значении истисна), `average_time_coefficient` — коэффициент, определяющий во сколько раз может быть превышено среднее время обновления.

Дополнительные защищенные атрибуты модели: `_log` — для хранения текущего лога (экземпляра `log_model`).

Модель предоставляет публичную функцию `get_log_model(cls, *args, **kwargs)`, которая возвращает атрибут `log_model` (функция добавлена для переопределения при необходимости).

Модель предоставляет публичную функцию `is_use_average_time(cls, *args, **kwargs)`, которая возвращает атрибут `use_average_time` (функция добавлена для переопределения при необходимости).

Модель предоставляет публичную функцию `get_average_time_coefficient(cls, *args, **kwargs)`, которая возвращает атрибут `average_time_coefficient` (функция добавлена для переопределения при необходимости).

Модель предоставляет публичную функцию `get_last_update_time(cls, filters, *args, **kwargs)`, которая возвращает из модели лога (`log_model`) последнее время завершения (`time_finished`) среди всех логов после применения фильтров (`filters`).

Модель предоставляет публичную функцию `get_last_update_time_attempt(cls, *args, **kwargs)`, которая возвращает результат функции `get_last_update_time` с фильтром на пропуск логов со значением истина в поле `was_canceled` (то есть время последней неотмененной попытки обновления).

Модель предоставляет публичную функцию `get_average_time_coefficient(cls, *args, **kwargs)`, которая возвращает результат функции `get_last_update_time` с фильтром на пропуск логов со значением истина в любом из полей `was_canceled`, `was_terminated`, `has_failed` (то есть время последнего успешного обновления).

Модель предоставляет публичную функцию `calc_average_update_time(cls, *args, **kwargs)`, которая вычисляет по имеющимся логам (записям модели из атрибута `log_model`), у которых непустые значения в полях `time_created` и `time_finished` и значение ложь в каждом из полей `was_created`, `was_terminated`, `has_failed`, то есть среднее время обновления всех успешных обновления. Функция полезна для того, чтобы избежать необходимости ручной

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

корректировки атрибута `time_limit` при постепенном увеличении времени обновления (например, если таблица растёт).

Модель переопределяет функцию `is_can_start`, которая в данной реализации возвращает истину, если 1) не имеется текущего обновления, или если 2) значение атрибута `use_average_time` — истина, и среднее время удалось рассчитать (имеется хотя бы 1 лог успешного обновления), и время текущего обновления (вычисляется при помощи атрибута `_time_started`) не превышает произведение среднего времени и значения атрибута `average_time_coefficient`, или 3) если среднее время обновления не было вычислено и время текущего обновления превышает значение атрибута `time_limit`. Иначе возвращается ложь. Данная реализация добавляет возможность прекращения зависшего обновления с использование среднего времени обновления, а не только предельного (`time_limit`).

Модель переопределяет функцию `_handle_cannot_start`, которая в данной реализации создает запись лога с истиной в поле `was_canceled` и датой завершения (`time_finished`), то есть сохраняет лог об отмене обновления.

Модель переопределяет функцию `_pre_update`, которая в данной реализации создает запись лога (`_log`) для фиксации момента начала обновления.

Модель переопределяет функцию `_post_update` которая в данной реализации сохраняет дату завершения (`time_finished`) в лог (`_log`), то есть фиксирует успешное окончание обновления.

Модель переопределяет функцию `_handle_exception`, которая в данной реализации сохраняет полное сообщение о возникшей ошибки в лог (`_log`), дату завершения (`time_finished`) и значение истина 1) в поле `was_terminated`, если было выброшено исключение о прерывании обновления, иначе — 2) в поле `has_failed` (то есть фиксирует либо ошибку обновления, либо его прерывание).

1.2.10. HistoryModel

Модель наследует `TimedModel` (для использования поля с временем создания записи).

Дополнительные публичные атрибуты модели: `objects` — для переопределения стандартного кверисета специальным (`HistoryQuerySet`), `group_by` — список названий полей для группировки при получении состояния строки на дату (определяющие уникальность

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

записи на дату), `date_field` — название поля-даты, по которому определяется состояние строки на дату.

Кверисет `HistoryQuerySet` предоставляет публичную функцию `for_date(self, group_by, date_field, date, *args, **kwargs)`, которая выбирает все записи модели, у которых в поле с названием `date_field` значение меньше или равно `date` (если `date` пусто, то этот фильтр не применяется), при этом группирует полученные строки по полям с названиями из списка `group_by` и из полученных групп выбирает записи с максимальным значением даты в поле с названием `date_field`. Таким образом, функция возвращает состояния строк на указанную дату.

Модель предоставляет публичную функцию `get_group_by(cls, *args, **kwargs)`, которая возвращает атрибут `group_by` (функция добавлена для переопределения при необходимости).

Модель предоставляет публичную функцию `get_date_field(cls, *args, **kwargs)`, которая возвращает атрибут `date_field`, если он не пуст, иначе — название поля `time_created`, которое является полем для отслеживания даты по умолчанию (функция добавлена для переопределения при необходимости).

Модель предоставляет публичную функцию `get_objects_for_date(cls, date, *args, **kwargs)`, которая возвращает результат вызова функции `for_date` кверисета `HistoryQuerySet`, передавая в нее аргументы `group_by`, `date_field` и `date`.

1.3. Условия, необходимые для выполнения программы

Состав требуемых технических средств: любое аппаратное обеспечение, на которое можно установить `python` [21] и сторонние библиотеки для него, которое имеет командный интерпретатор и к которому подключены источник ввода и источник графического вывода.

Состав требуемых программных средств: `Python 3.10` или новее [21], `Django 4.1.6` или новее [14].

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

2. ХАРАКТЕРИСТИКА ПРОГРАММЫ

Режим работы программы соответствует стандартному режиму работы пакетов средств разработки (SDK) для языка Python [21].

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

3. ОБРАЩЕНИЕ К ПРОГРАММЕ

Обращение к программе соответствует стандартному режиму работы пакетов средств разработки (SDK) для языка Python [21].

Подробное описание имеющихся функций программы приведено в пункте 1.2 настоящего документа.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

4. ВХОДНЫЕ И ВЫХОДНЫЕ ДАННЫЕ

Подробное описание входных данных для используемых функций (аргументов) и выходных данных (возвращаемых ими значений) приведено в пункте 1.2 настоящего документа.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

5. СООБЩЕНИЯ

Программа не предоставляет сообщений.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

1. ГОСТ 19.101-77 Виды программ и программных документов. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
2. ГОСТ 19.102-77 Стадии разработки. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
3. ГОСТ 19.103-77 Обозначения программ и программных документов. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
4. ГОСТ 19.104-78 Основные надписи. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
5. ГОСТ 19.105-78 Общие требования к программным документам. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
6. ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
7. ГОСТ 19.201-78 Техническое задание. Требования к содержанию и оформлению. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
8. ГОСТ 19.603-78 Общие правила внесения изменений. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
9. ГОСТ 19.604-78 Правила внесения изменений в программные документы, выполненные печатным способом. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
10. ГОСТ 19.404-79 Пояснительная записка. Требования к содержанию и оформлению. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
11. ГОСТ 19.301-79 Программа и методика испытаний. Требования к содержанию и оформлению. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
12. ГОСТ 19.401-78 Текст программы. Требования к содержанию и оформлению. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
13. ГОСТ 19.504-79 Руководство программиста. Требования к содержанию и оформлению. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
14. Django // [Электронный ресурс] — Режим доступа: свободный, URL: <https://www.djangoproject.com/> (04.02.2022)
15. Wikipedia // [Электронный ресурс] — Режим доступа: свободный, URL: <https://ru.wikipedia.org/wiki/SDK> (04.02.2022)
16. Wikipedia // [Электронный ресурс] — Режим доступа: свободный, URL: https://ru.wikipedia.org/wiki/Модель_данных (04.02.2022)
17. Wikipedia // [Электронный ресурс] — Режим доступа: свободный, URL: https://ru.wikipedia.org/wiki/Программное_обеспечение (04.02.2022)
18. Wikipedia // [Электронный ресурс] — Режим доступа: свободный, URL: <https://ru.wikipedia.org/wiki/ORM> (04.02.2022)

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

19. Wikipedia // [Электронный ресурс] — Режим доступа: свободный, URL:
https://ru.wikipedia.org/wiki/База_данных
(04.02.2022)
20. Wikipedia // [Электронный ресурс] — Режим доступа: свободный, URL:
https://ru.wikipedia.org/wiki/Объектно-ориентированное_программирование
(04.02.2022)
21. Python // [Электронный ресурс] — Режим доступа: свободный, URL:
<https://www.python.org/>
(04.02.2022)

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

Лист регистрации изменений

[illegible]