Добрый день. Меня зовут Мишин Степан, я являюсь студентом группы 09-051. Темой моей выпускной квалификационной работы является «Разработка системы контроля версий».

**Цели и задачи**

Целью данной дипломной работы является разработка и реализация собственной системы контроля версий (VCS) с учетом современных требований в области разработки программного обеспечения. Основной акцент будет сделан на создании эффективных механизмов управления изменениями, обеспечении совместной работы множества разработчиков, а также предоставлении надежных инструментов для отслеживания и анализа истории изменений в проекте.

В ходе выполнения дипломной работы планируется провести анализ существующих систем контроля версий, выявить их преимущества и недостатки, и на основе полученных данных разработать и реализовать инновационное решение. Дополнительными задачами будут тестирование системы, обеспечение безопасности данных и создание документации для пользователя. Работа направлена на создание гибкой и эффективной инструментальной базы для разработчиков, способной успешно конкурировать в современной среде программной инженерии.

**Анализ предметной области**

В современном мире разработка программного обеспечения становится все более сложной и коллаборативной задачей, требующей эффективного управления изменениями в исходном коде. Системы контроля версий (VCS) играют ключевую роль в этом процессе, предоставляя механизмы отслеживания, управления и совместной работы над версиями программного продукта.

Тема разработки новой системы контроля версий становится важной, учитывая постоянное развитие технологий и появление новых требований к процессам разработки. Перспективы улучшения эффективности совместной работы, безопасности данных и интеграции с современными средами разработки делают актуальной не только оптимизацию уже существующих VCS, но и создание новых, инновационных систем.

**Обзор аналогов**

Одним из ведущих инструментов в этой области является Git, распределенная система контроля версий, разработанная для обеспечения высокой производительности и гибкости при работе с различными типами проектов. Git стал стандартом в индустрии благодаря своей скорости, надежности и обширной поддержке со стороны сообщества разработчиков.

Однако, помимо Git, на рынке существует ряд других систем контроля версий, каждая из которых обладает своими уникальными особенностями и предназначена для решения конкретных задач. Mercurial, SVN, Perforce и другие предоставляют различные модели работы, интеграции и протоколы доступа, удовлетворяя разнообразные потребности команд разработки.

**Техническое задание**

При анализе существующих решений были выявлены функциональные и нефункциональные требования для разрабатываемой VCS.

Функциональные требования включают в себя: инструменты управления репозиториями, работа с ветками, отслеживание изменений файлов, работа с коммитами, совместная работа разработчиков, разрешение конфликтов при слиянии, аутентификация и авторизация, командный и визуальный интерфейс, разработка сервера для VCS, где также можно будет управлять версиями и отслеживать изменения файлов.

Нефункциональные требования следующие: удобство использования, надежность и безопасность, скорость работы, минимальные системные требования, наличие документации, поддержка кроссплатформенности.

**Инструменты разработки**

В качестве основного языка программирования я выбрал Python, т.к. имею опыт разработки на нем и этот язык обладает большим набором библиотек и инструментов, которые помогут при решении задач, возникающих в ходе разработки. Celery нужен для поддержки фоновых или асинхронных задач. Gunicorn будет использоваться в качестве сервера. PyQt для создания пользовательского интерфейса системы. Django – фреймворк для создания веб приложений пригодится для создания сервера для VCS, PostgreSQL – БД для хранения данных для сервера VCS, MongoDB – БД для хранения изменений файлов, pytest – для тестирования системы. Библиотека Libgit2 или Pygit2 может быть полезной если будет необходимость создавать VCS, совместимую с Git.

Разработка VCS будет направлена на решение текущих вызовов в области разработки программного обеспечения, обеспечение более эффективного управления версиями кода, а также для повышения удобства и простоты использования системами контроля версий в динамичной и конкурентной среде разработки.