Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут ім. Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

Розрахункова графічна робота
З дисципліни «Інтеграційні програмні системи»

Виконали:
Студенти 4 курсу ФІОТ
Групи ІО-51
Зубар Ілля Андрійович
Тарановський Микола Володимирович
Швець Андрій Валентинович
Бригада КІА

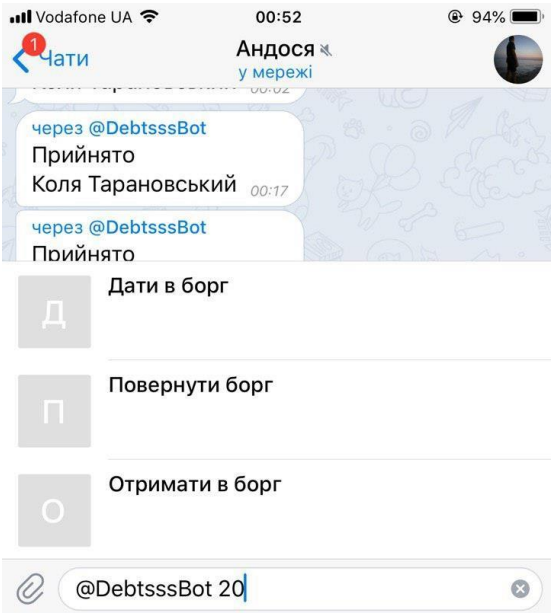
Київ
2018

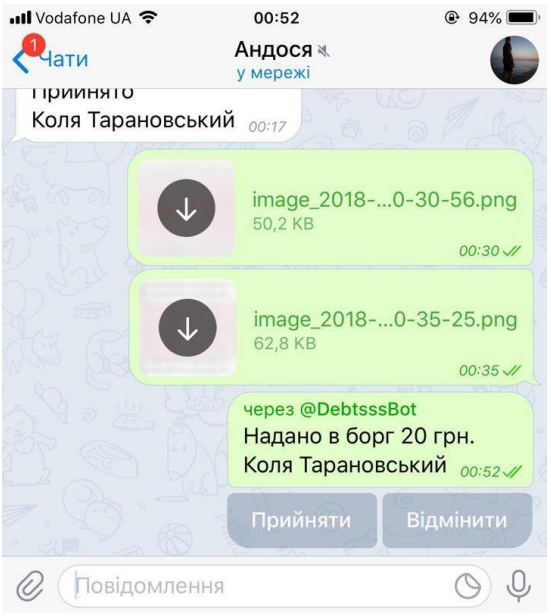
Короткий опис проекту

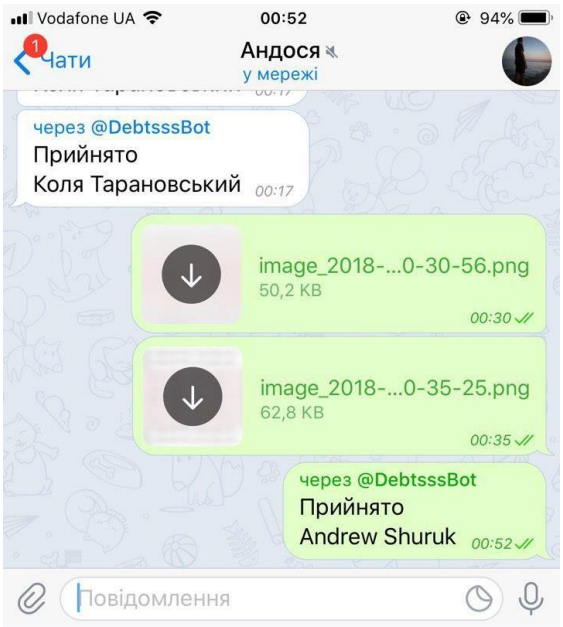
Телеграм бот призначений зберігати інформацію про проведені фінансові операції (борги, кредити) між 2 користувачами.

Сценарії роботи системи

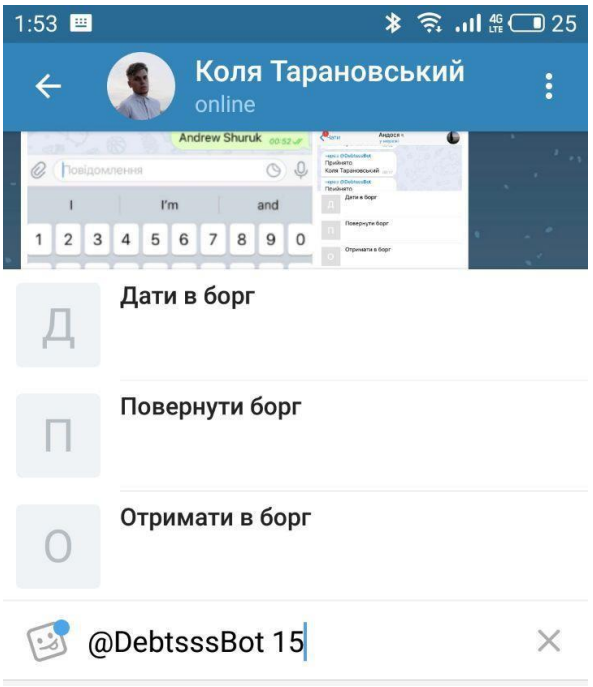
1. Кредитор надає кошти боржнику
- кредитор викликає бота і вводить необхідну суму
- обирає операцію ‘Дати в борг’
- боржник натискає кнопку прийняти
- користувачі перевіряють свій фінансовий журнал командою ‘/myWallet` у діалозі з ботом

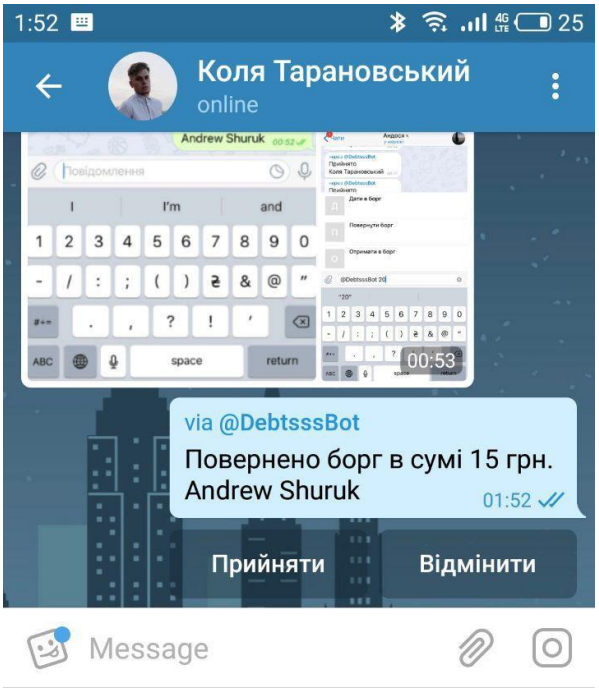


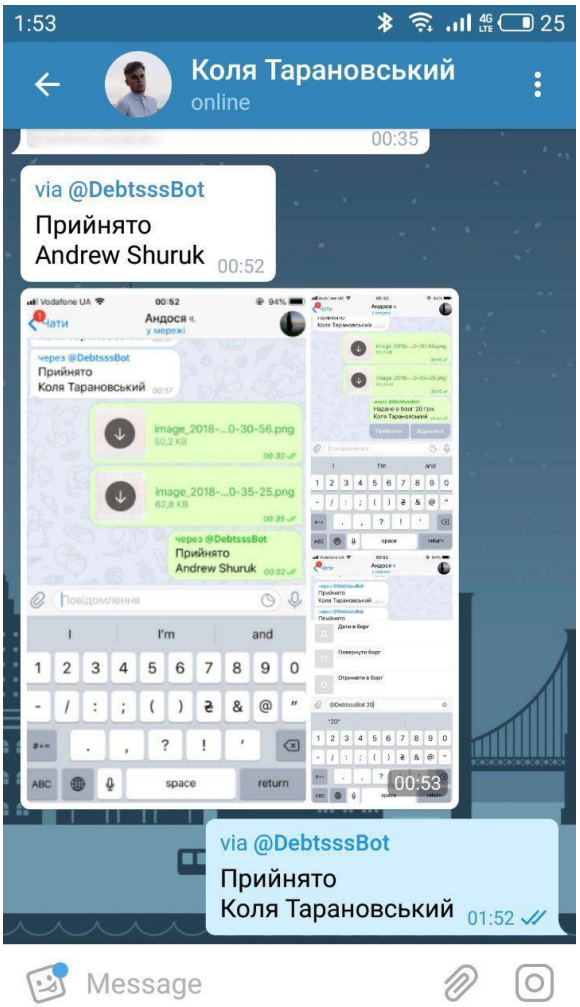


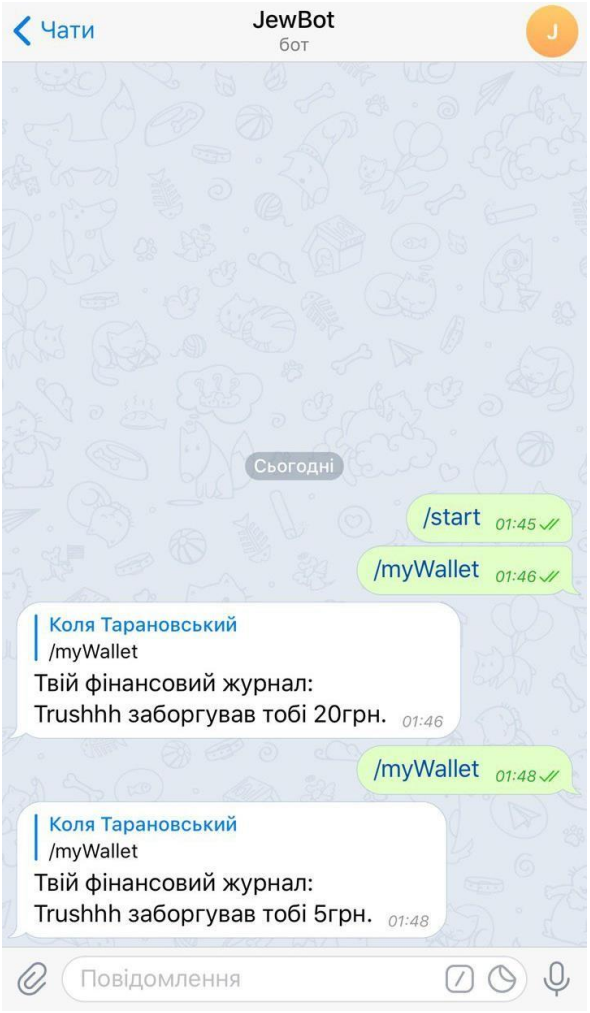


2. Боржник повертає кошти кредитору
- боржник викликає бота і вводить необхідну суму
- обирає операцію ‘Повернути борг’
- кредитор натискає кнопку прийняти
- користувачі перевіряють свій фінансовий журнал командою ‘/myWallet` у діалозі з ботом

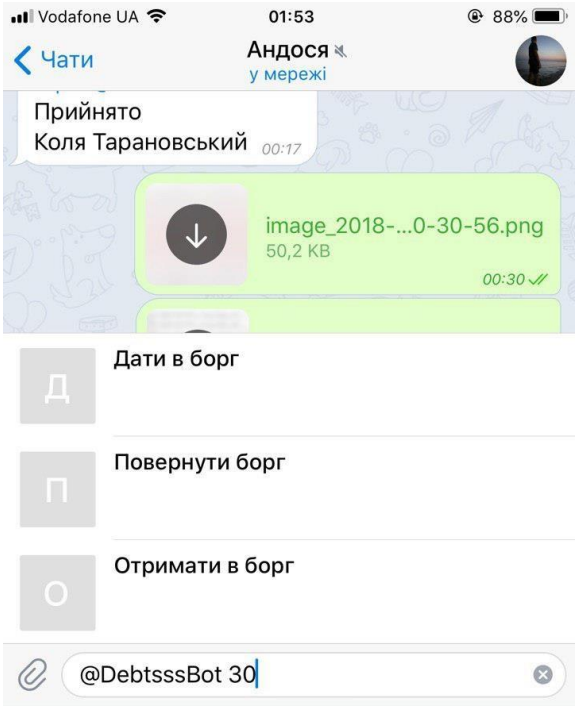


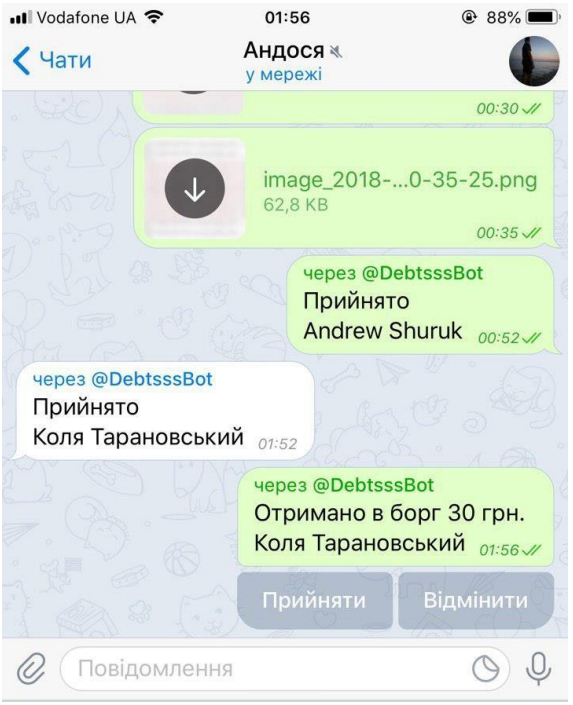


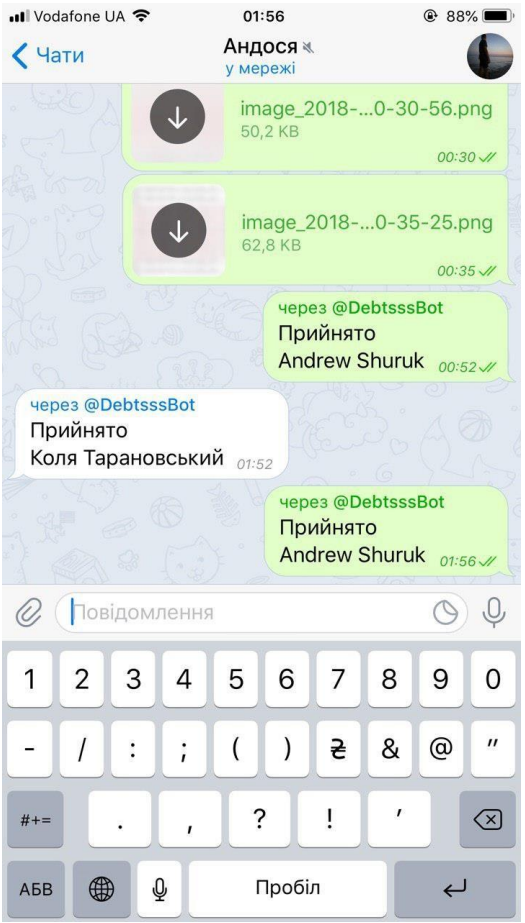


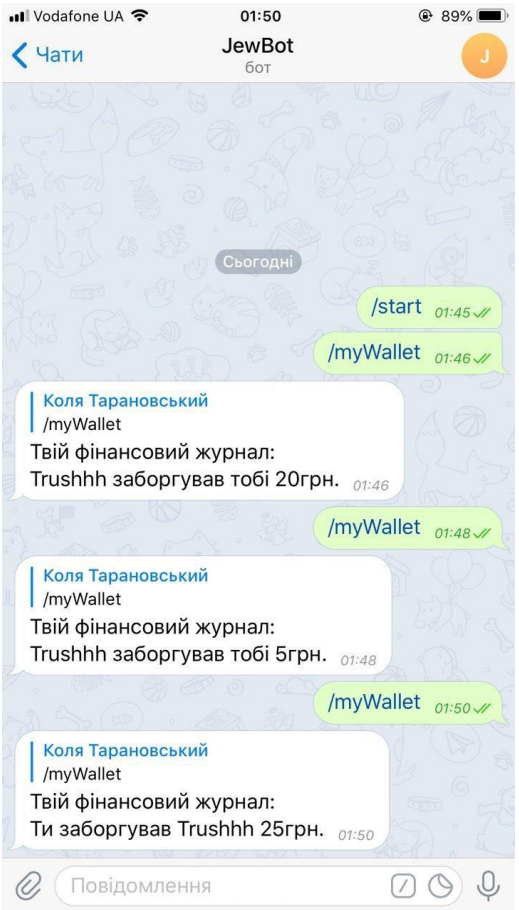


3. Боржник отримує кошти в борг
- боржник викликає бота і вводить необхідну суму
- обирає операцію ‘Отримати в борг’
- кредитор натискає кнопку прийняти
- користувачі перевіряють свій фінансовий журнал командою ‘/myWallet` у діалозі з ботом









Система збірки, що використовується у проекті

PyInstaller – система збірки, що зчитує написаний Python скрипт. Він аналізує код, щоб виявити всі модулі та бібліотеки, які скрипт потребує для виконання. Потім він збирає копії всіх цих файлів - у тому числі активний інтерпретатор Python і поміщає їх зі скриптом в одній папці чи в одному виконуваному файлі. PyInstaller не є крос-компільованою системою. Для створення програми для Windows необхідно запустити PyInstaller у Windows; щоб створити Linux-додаток, запускати його в Linux і т.д.

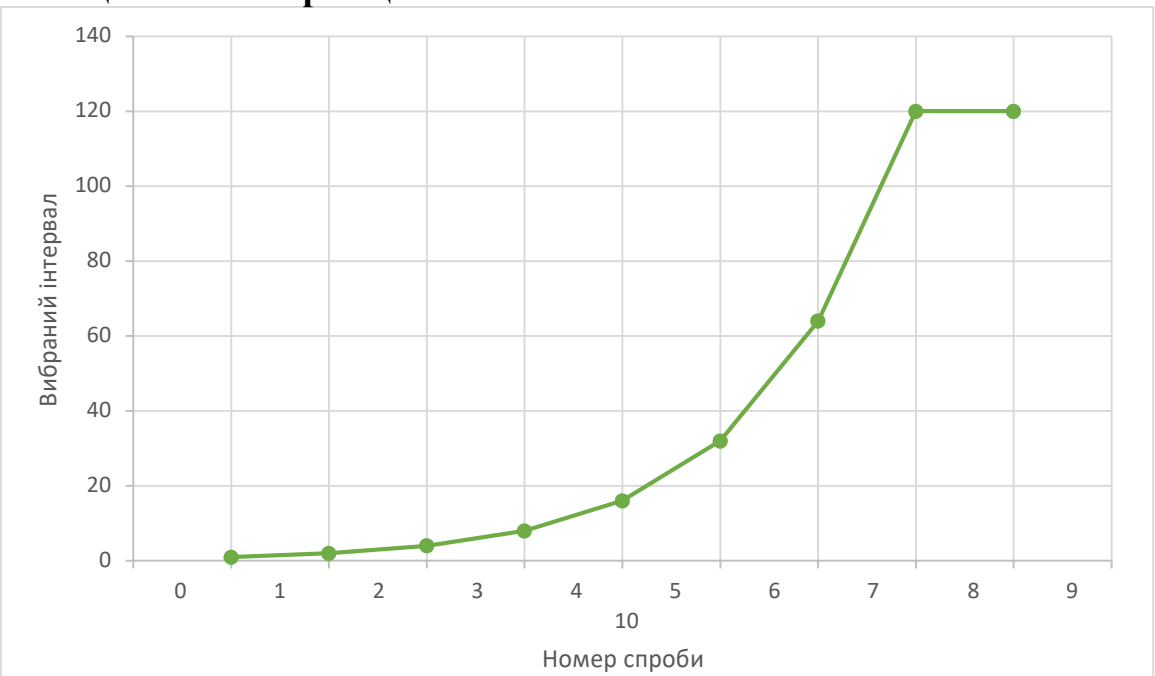
Основні переваги

- Працює з будь-якою версією Python 2.7 / 3.4-3.7.
- Повністю мультиплатформенний і використовує підтримку ОС для завантаження динамічних бібліотек, забезпечуючи таким чином повну сумісність.
- Правильно пов'язує основні Python packages, такі як numpy, PyQt4, PyQt5, PySide, Django, wxPython, matplotlib та інші.
- Сумісний з багатьма пакетами сторонніх виробників. (Всі необхідні трюки, щоб зробити роботу зовнішніх пакетів вже інтегровані.)

Перелік та опис задач, які виконуються на сервері безперервної інтеграції.

- Build образу проекту. Збірка проекту.
  ```
  docker build -t bot .
  ```
- Запуск Docker контейнера з БД MongoDB на локальному хості
  ```
  docker run -d --rm -p 27017:27017 --name mongodb mongo
  ```
- Виконання прямого тестування коду, тестування поведінки за допомогою моків, інтеграційних тестів за допомогою Docker, сформовані у вигляді юніт тестів.
  ```
  python -m unittest discover
  ```
- Запуск аналізатора коду
  ```
  pylint main.py
  ```

Графік, який ілюструє вибрані інтервали для повтору спроб при експоненціальній витримці.



Випадок з помилкою БД.

- БД спочатку функціонує правильно.
- Потім вона вимикається.
- Клієнт виконує спроби повтору.
- Коли БД вмикається, то система функціонує знову правильно

```
{'_id': ObjectId('5c23fb972c6fc31a042d624a'), 'user_id': 310120198, 'debts':
 [{'partner_id': 332915402, 'debt': 155, 'data': datetime.datetime(2018, 12, 27, 0, 17,
 23, 239000)}]}
{'_id': ObjectId('5c23fb972c6fc31a042d624b'), 'user_id': 332915402, 'debts':
 [{'partner_id': 310120198, 'debt': -155, 'data': datetime.datetime(2018, 12, 27, 0, 17,
 23, 243000)}]}
WARNING:root:PyMongo auto-reconnecting... localhost:27017: [WinError 10061] No connection
 could be made because the target machine actively refused it. Waiting 1.0 seconds.
WARNING:root:PyMongo auto-reconnecting... localhost:27017: [WinError 10061] No connection
  could be made because the target machine actively refused it. Waiting 2.0 seconds.
WARNING:root:PyMongo auto-reconnecting... localhost:27017: [WinError 10061] No connection
  could be made because the target machine actively refused it. Waiting 4.0 seconds.
WARNING:root:PyMongo auto-reconnecting... localhost:27017: [WinError 10061] No connection
  could be made because the target machine actively refused it. Waiting 8.0 seconds.
WARNING:root:PyMongo auto-reconnecting... localhost:27017: [WinError 10061] No connection
  could be made because the target machine actively refused it. Waiting 16.0 seconds.
WARNING:root:PyMongo auto-reconnecting... localhost:27017: [WinError 10061] No connection
  could be made because the target machine actively refused it. Waiting 32.0 seconds.
WARNING:root:PyMongo auto-reconnecting... localhost:27017: [WinError 10061] No connection
  could be made because the target machine actively refused it. Waiting 64.0 seconds.
WARNING:root:PyMongo auto-reconnecting... localhost:27017: [WinError 10061] No connection
  could be made because the target machine actively refused it. Waiting 120.0 seconds.
WARNING:root:PyMongo auto-reconnecting... localhost:27017: [WinError 10061] No connection
  could be made because the target machine actively refused it. Waiting 120.0 seconds.
{'_id': ObjectId('5c2400942c6fc316949ce903'), 'user_id': 310120198, 'debts':
 [{'partner_id': 332915402, 'debt': 200, 'data': datetime.datetime(2018, 12, 27, 0, 28,
 36, 905000)}]}
{'_id': ObjectId('5c2400942c6fc316949ce904'), 'user_id': 332915402, 'debts':
 [{'partner_id': 310120198, 'debt': -200, 'data': datetime.datetime(2018, 12, 27, 0, 28,
 36, 908000)}]}
```