**Explanatory note**

Program/Project name: “Telegram Bot: Learning Assistant”.

The aim of the project has been the development of a Telegram Bot with the capability to remind users of upcoming deadlines and home tasks scheduled for practical sessions. The app is essentially split into two parts: a control panel and a module responsible for the bot operation, run in parallel.

All information is stored in Microsoft SQL database (for the simplicity, we have used the localdb server). A GUI is built on WPF utilising MVVM pattern.

One may download project files from the GitHub repository (remember to specify the “master” branch): <https://github.com/koxrel/LearningAssistantBot>.

The team consists of two members:

* Igor Tresoumov (responsible for API Interaction, Database implementation and Data Access Layer);
* Sergey Pavlov (responsible for GUI and user interaction with model classes).

Unit-tests for classes are written if the member responsible deems it necessary (that’s why tests for database and API interactions were not designed since they rely on external sources).

**Classes description**

Project LearningAssistant:

* Command — implementation of ICommand interface;
* Factory — a unified place for initializing dependencies;
* TextToColourConverter — implementation of IValueConverter interface, converts string status to colour;
* Navigator — implementation of INavigator interface, navigates between windows;
* AdditionalViewModel — a view model for Additional window;
* DetailsBaseViewModel — a base view model for windows intended to display database tables;
* DeadlineExplorerViewModel — a view model for DeadlineExplorer window;
* HoTaExplorerViewModel — a view model for HoTaExplorer window;
* UserExplorerViewModel — a view model for UserExplorer window;
* ViewModel — a view model for MainWindow window.

Project LearningAssistant.Database:

* Context — a database access provider;
* TextBuilder — represents classes as text;
* DataAccess — implementation of IDataAccess, provides method for interacting with a database;
* Deadline, Hometask, User — entity classes, implement IDeadline, IHometask, IUser respectively.

Project LearningAssistant.TelegramBot:

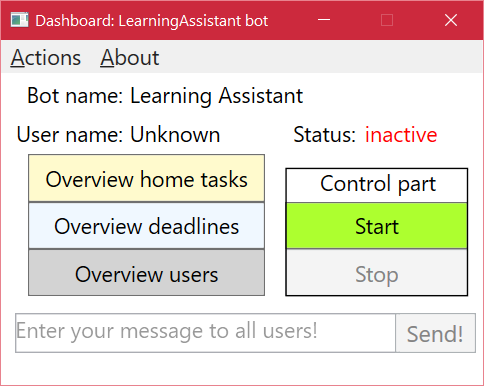
* BotWebRequest — implementation of IBotWebRequest, provides functionality required to run a bot;
* Factory — a unified place for initializing dependencies;
* Chat, Message, Updates, Update, User — DTO classes.

Project LearningAssistant.Tests:

* TextBuilderTests — tests for TextBuilder class;
* TextToColourConverterTests — tests for TextToColourConverter class.

**User Interface**

**Control Panel:**



Pretty self-explanatory.

Start/Stop buttons — start/stop a bot’s activity.

Bot name — displays the name of the bot after the Start button is clicked.

User name — a user name by which a bot can be accessed in Telegram.

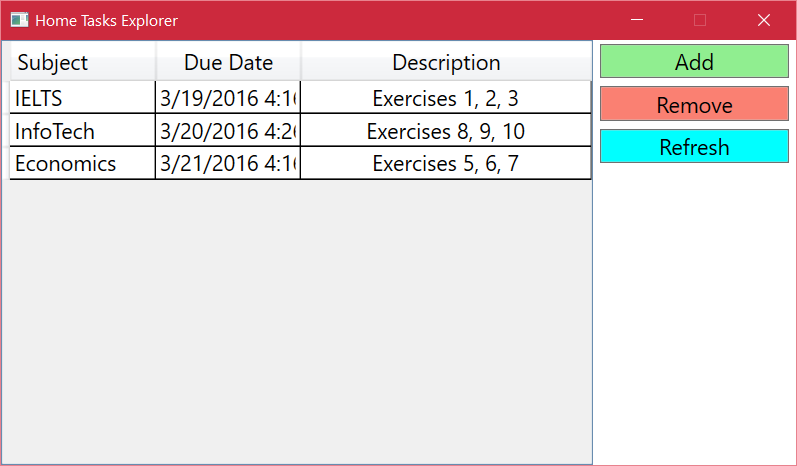
Send button — send a message (specified on the textbox) to all users who have reached a bot at least once.

Overview home tasks/deadlines/users buttons — display related information in a separate window.

Actions menu — all possible actions are displayed there.

About – Info — display information about the application.

**Home Tasks Explorer Window:**



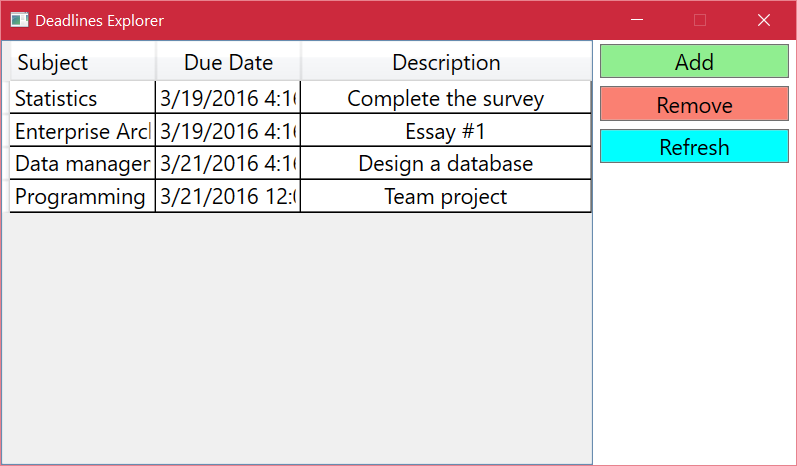
A table with available home tasks is displayed.

Add — calls a separate window for adding either a home task or a deadline to a database.

Remove — remove selected item from a database.

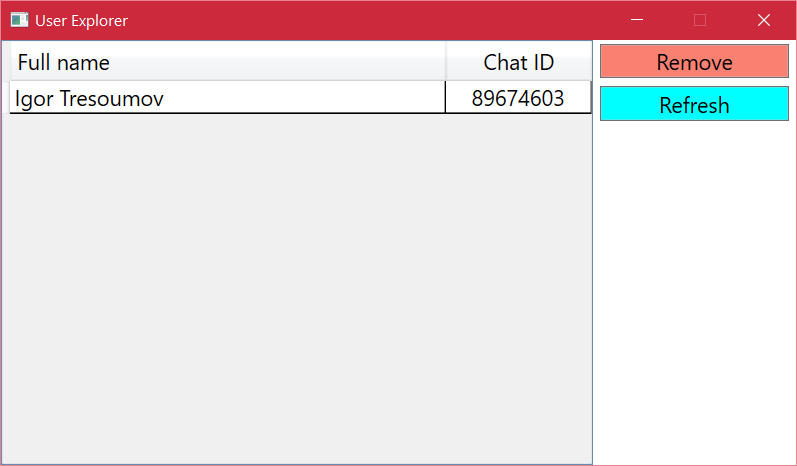
Refresh — refresh elements in the table.

**Deadlines Explorer Window:**



All remarks for Home Tasks Explorer Window are relevant here, except for the fact it works with deadlines.

**User Explorer Window:**

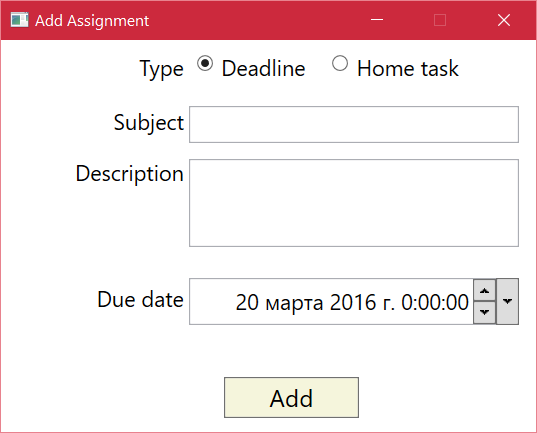


View all the users who have accessed a bot.

Remove — remove selected item from a database.

Refresh — refresh elements in the table.

**Add Assignment Window:**



Type — an assignment will be saved using the type provided.

Subject and Description are required text fields (user provides information).

Due date — a control for choosing date and time of an assignment (DateTimePicker control from Extended WPF Toolkit library).

Add — save an assignment in a database.