```
/*
 ********************************************************************************
 * Universidade de Brasília
 * Instituto de Ciências Exatas
 * Departamento de Ciência da Computação
 * Matéria: Programação Sistemática
 * Professor: Fernando Albuquerque
 * Aluno: Aaron Sue
 * Matrícula: 13/0020249
 * Trabalho Prático
 * Relação de interfaces com as respectivas assinaturas de funções
 ********************************************************************************
 */

Interface 1: Estruturas de dados
    struct desenvolvedor;
    struct produto;
    struct defeito;

    void get_string(char *str, size_t size);
    void get_password(char *pass, size_t size);
    int  validate_email(const char *email);
    int  validate_name(const char *nome);
    int  validate_code(const char *cod);
    int  validate_version(const char *versao);

Interface 2: Interface do módulo de persistência de desenvolvedores
    int register_developer(const struct desenvolvedor *dev);
    int read_developer(struct desenvolvedor *dev);
    int rewrite_developer(const struct desenvolvedor *dev);
    int delete_developer(struct desenvolvedor *dev);
    int count_developers(void);

Interface 3: Interface do módulo de persistência de produtos
    int register_product(const struct produto *prod);
    int read_product(struct produto *prod);
    int rewrite_product(const struct produto *prod);
    int delete_product(struct produto *prod);
    int count_products(void);
    int remove_product_leader(char *dev_email);

Interface 4: Interface do módulo de persistência de defeitos
    int register_defect(const struct defeito *bug);
    int read_defect(struct defeito *bug);
    int rewrite_defect(const struct defeito *bug);
    int delete_defect(struct defeito *bug);
    int count_defects(void);
    int remove_product_defects(char *cod_prod);

Interface 5: Interface da lógica do negócio
    int check_login(struct desenvolvedor *dev);
    int register_new_developer(struct desenvolvedor *dev);
    int number_of_developers(void);
    int load_developer(struct desenvolvedor *dev);
    int overwrite_developer(const struct desenvolvedor *dev);
    int remove_developer(struct desenvolvedor *dev);
    int return_profile(const struct desenvolvedor *dev);
    int assign_new_project_leader(struct desenvolvedor *dev,
                                          const char *dev_email);

    int register_new_product(struct produto *prod);
    int load_product(struct produto *prod);
    int overwrite_product(const struct produto *prod);
    int remove_product(struct produto *prod);
    int assign_product_leader(char *dev_email, char *prod_cod);

    int register_new_defect(struct defeito *bug);
    int load_defect(struct defeito *bug);
```

```
    int overwrite_defect(const struct defeito *bug);
    int associate_defect(struct desenvolvedor *dev, char *cod_def);
    int assign_defect_to_developer(char *dev_email, char *def_cod);

Interface Auxiliar: Estruturas
    struct desenvolvedor {
        char   nome[NAME_SIZE];
        char   email[EMAIL_SIZE];
        char   senha[PASS_SIZE];
        size_t lid_proj;
        size_t lid_prod;
        size_t cand_def;
        char   cand1[CODE_SIZE];
        char   cand2[CODE_SIZE];
        size_t sol_def;
        char   def[CODE_SIZE];
        size_t excluido;
    };

    struct produto {
        char   nome[NAME_SIZE];
        char   cod[CODE_SIZE];
        char   versao[VERSION_SIZE];
        char   lider[EMAIL_SIZE];
        size_t excluido;
    };

    struct defeito {
        char   cod[CODE_SIZE];
        char   desc[DESC_SIZE];
        size_t est;
        size_t votos;
        char   dt_ab[DATE_SIZE];
        char   dt_fc[DATE_SIZE];
        char   des_sel[EMAIL_SIZE];
        char   prod[CODE_SIZE];
        size_t excluido;
    };

    void get_string(char *str, size_t size);
    void get_password(char *pass, size_t size);
    int  validate_email(const char *email);
    int  validate_name(const char *nome);
    int  validate_code(const char *cod);
    int  validate_version(const char *versao);
```